AF/2152
SFW

920584-906003

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In the application of	:	O' Doherty, M.
Serial No.	:	09/520,853
Filed	:	December 22, 1999
For	:	Java Enhanced SIP
Examiner	:	Flynn, Kimberly D.
Art Unit	:	2152
Customer number	:	23644

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450," on November 22, 2005
Name of person signing Minnie Wilson
Signature Minnie Wilson

APPEAL BRIEF

Honorable Director of Patents and Trademarks
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

This appeal is from the Examiner's final Office Action mailed May 6, 2005 and the Advisory Action mailed August 29, 2005 in which all pending claims (namely Claims 1-33) were rejected. A timely Notice of Appeal was filed with the required fee.

This brief is being filed along with the required $500 fee pursuant to 37 C. F. R. § 41.20(b)(2).

11/28/2005 HGUTEMA1 00000032 09520853
01 FC:1402 500.00 OP

(i) Real Party in Interest

This application is assigned to Nortel Networks Limited. The assignments are recorded at Reel 010633, Frame 0187.

(ii) Related Appeals and Interferences

There are no related appeals or interferences.

(iii) Status of Claims

This application was filed with claims 1 to 33. In the responses of December 23, 2004, June 10, 2004, June 4, 2003 and January 20, 2003 claims 1, 3, 8, 10, 13, 15-24, 26, 27, and 30-33 were each amended at least once. The remaining claims are as originally filed.

(iv) Status of Amendments

Claims 1-33 remain pending and have all been considered by the Examiner and finally rejected. The Examiner's response was in the Advisory Action mailed August 29, 2005. There have been no amendments of the claims following the final rejection, but a response of July 6, 2005 was filed to argue the rejections.

(v) Summary of Claimed Subject Matter

The invention as presently claimed is concerned with using an improved Session Initiation Protocol (SIP) to transfer executable code from a first node to a second node of a communications network. SIP is defined by IETF RFC 2543 as an application layer control (ie signaling) protocol for creating, modifying and terminating sessions such as Internet multimedia conferences, Internet telephone calls and multimedia distribution (see IETF RFC 2543 abstract set out below).

> "The Session Initiation Protocol (SIP) is an application-layer control (signaling) protocol for creating, modifying and terminating sessions with one or more participants. These sessions include Internet multimedia conferences, Internet telephone calls and multimedia distribution …
>
> SIP invitations used to create sessions carry session descriptions which allow participants to agree on a set of compatible media types. … SIP is designed to be independent of the lower-layer transport protocol and can be extended with additional capabilities."

It is well-known that SIP operates at Layer 5 (the Session Layer) of the Open Systems Interconnect (OSI) seven layer model of the network protocol stack. Layer 5 (the Session Layer) is immediately above Layer 4 (the Transport Layer) and 2 layers below Layer 7 (the Application Layer). See table below which, for convenience of review, is all on the following page:

Application (Layer 7)	This layer supports application and http://www.webopedia.com/quick_ref/end_user.htmlend-user processes. Everything at this layer is application-specific. This layer provides application services for file transfers, e-mail, and other network software services.
Presentation (Layer 6)	This layer provides independence from differences in data representation by translating from application to network format, and vice versa.
Session (Layer 5)	This layer establishes, manages and terminates connections between applications.
Transport (Layer 4)	This layer provides transfer of data between end systems, or hosts, and is responsible for end-to-end error recovery and flow control. It ensures complete data transfer.
Network (Layer 3)	This layer provides switching and routing for transmitting data from node to node. Routing and forwarding are functions of this layer, as well as addressing, internetworking, error handling, congestion control and packet sequencing.
Data Link (Layer 2)	At this layer, data packets are encoded and decoded into bits. It furnishes transmission protocol knowledge and management and handles errors in the physical layer, flow control and frame synchronization.
Physical (Layer 1)	This layer conveys the bit stream - electrical impulse, light or radio signal -- through the network at the electrical and mechanical level. It provides the hardware means of sending and receiving data on a carrier..

As defined by IETF RFC 2543, SIP uses two types of message – request messages and response messages – according to a client/server model. Request messages include INVITE, ACK, OPTIONS, BYE, CANCEL and REGISTER messages. INVITE, for example, is used to invite an application (such as an Internet phone client) to a session. Response messages indicate the success or otherwise of an attempt to understand and satisfy a request message. Response messages use a status code to indicate the outcome. For example, 200 OK indicates that the request was understood and successfully acted upon.

As a signaling protocol, SIP was not designed to be used to transfer user data. SIP was designed to be used to create, modify and terminate sessions in which other protocols (such as the Real Time Protocol (RTP)) may transfer user data, such as voice, video and multimedia. As such, although the INVITE, ACK, OPTIONS, an REGISTER messages may contain message bodies, the message bodies is always a session description – ie a description of the session which is being set up, modified or terminates, for example, whether it contains voice only or video, at what resolution, speech quality, which codecs to use and so on. See IETF RFC 2543 section 8.1 set out below. Typically session description information is provided using the Session Description Protocol (SDP).

> "Requests MAY contain message bodies unless otherwise noted. Within this specification, the BYE request MUST NOT contain a message body. For ACK, INVITE and OPTIONS, the message body is always a session description. The use of message bodies for REGISTER requests is for further study."

Despite this clear teaching to the contrary, the present inventor realized that it would nonetheless be useful to store data other than a session description in a SIP message. In particular, the inventor realized that it would be useful to store computer software code in a SIP message, such as an INVITE request message. According to the present invention, computer software code can either be stored in a SIP message by adding it to the SIP message or by adding an address to the SIP message which indicates where the computer software code is stored.

The inventor realized that multimedia conferencing (and other conference calls) are a relatively complicated service for an end user to use. By storing computer software code in a SIP message, which may be used to invite the user's application to the conference, the end user can execute the code at his or her computing device to automatically set up the conference, for example. This is just one example of the type of extra functionality that may be provided by storing computer software code in a SIP message.

It is important to note that, by suggesting that computer software code should be stored in a SIP message, the inventor did something surprising and quite contrary to the industry standard for SIP (ie contrary to IETF RFC 2543 itself). Moreover, given that SIP is a signalling protocol (not a data transfer protocol), it is even more surprising to use a SIP message to store and thereby transfer computer software code.

(vi) Grounds of Rejection To Be Reviewed on Appeal

There are three rejections at issue:

1. the rejection of claims 1, 2, 5, 9, 10-15 and 20-26 under 35 USC §103(a) as being obvious over Handley (IETF RFC 2543 discussed above) in view of Venkatraman (US Patent Number 6,014,688);

2. the rejection of claims 3, 4, 7 and 8 under 35 USC §103(a) as being obvious over Handley in view of Venkatraman and further in view of Gampper (US Patent number 6,003,082); and

3. the rejection of claims 14, 15 and 22 under 35 USC §103(a) as being obvious over Handley in view of Venkatraman and further in view of Gampper (US Patent number 6,003,082).

The Examiner has not provided any argument in the Final Office Action dated May 6, 2005 why claims 27 to 33 are still rejected, but it can be assumed that the reasons are similar to those set out in the above claim rejections.

(vii) Argument

Grounds 1-3:

Independent claim 1 is directed to a method of transferring computer software code between a first and a second node in a communications network, each of said nodes comprising a SIP client. The claim includes the following features:

(i) storing computer software code in a SIP message;
(ii) sending the SIP message and computer software code from the first SIP client associated with the first node to the second SIP client associated with the second node; and
(iii) executing the computer software using the second node.

In *ex parte* examination of patent applications, the Patent and Trademark Office bears the burden of establishing a *prima facie* case of obviousness. MPEP § 2142; *In re Fritch*, 972 F.2d 1260, 1262, 23 U.S.P.Q.2d 1780, 1783 (Fed. Cir. 1992). The initial burden of establishing a *prima facie* basis to deny patentability to a claimed invention is always upon the Patent and Trademark Office. MPEP § 2142; *In re Oetiker*, 977 F.2d 1443, 1445, 24 U.S.P.Q.2d 1443, 1444 (Fed. Cir. 1992); *In re Piasecki*, 745 F.2d 1468, 1472, 223 U.S.P.Q. 785, 788 (Fed. Cir. 1984). Only when a *prima facie* case of obviousness is established does the burden shift to the applicant to produce evidence of non-obviousness. MPEP § 2142; *In re Oetiker*, 977 F.2d 1443, 1445, 24 U.S.P.Q.2d 1443, 1444 (Fed. Cir. 1992); *In re Rijckaert*, 9 F.3d 1531, 1532, 28 U.S.P.Q.2d 1955, 1956 (Fed. Cir. 1993). If the Patent and Trademark Office does not produce a *prima facie* case of unpatentability, then without more the applicant is entitled to grant of a patent. *In re Oetiker*, 977 F.2d 1443, 1445, 24 U.S.P.Q.2d 1443, 1444 (Fed. Cir. 1992); *In re Grabiak*, 769 F.2d 729, 733, 226 U.S.P.Q. 870, 873 (Fed. Cir. 1985). A *prima facie* case of obviousness is established when the teachings of the prior art itself suggest the

claimed subject matter to a person of ordinary skill in the art. *In re Bell*, 991 F.2d 781, 783, 26 U.S.P.Q.2d 1529, 1531 (Fed. Cir. 1993).

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed invention and the reasonable expectation of success must both be found in the prior art, and not based on applicant's disclosure. MPEP § 2142.

First, There is No Motivation to Combine

Appellant denies that one skilled in the art would be motivated to modify the teachings of Handley (RFC 2543) to include computer software code in SIP messages as alleged by the Examiner.

As discussed above, Handley is the industry standard on SIP. Handley not only fails to disclose the possibility of storing computer software code in a SIP message, as admitted by the Examiner, it actually teaches away from so doing - see section 8.1 set out below:

> "Requests MAY contain message bodies unless otherwise noted. Within this specification, the BYE request MUST NOT contain a message body. For ACK, INVITE and OPTIONS, the message body is **always** a session description. The use of message bodies for REGISTER requests is for further study." (emphasis added)

Thus, Appellant believes that one skilled in the art, but having no inventive capacity, would not even consider modifying Handley as alleged by the Examiner with perfect hindsight.

Furthermore, the Examiner argues that "it would have been obvious to a person having ordinary skill in the art to modify the SIP: Session Initiation Protocol taught by Handley to include the embedded software code taught by Venkatraman in order to enable the implementation of enhanced services not provided for by the native SIP protocol".

Venkatraman teaches an enhanced email system. In one embodiment of Venkatraman, executable software ("recipient executable software") may be embedded in a conventional email message so that a recipient computer may present enhanced content (which is also embedded in the email) to a user – see column 3 line 66 to column 4 line 12 set out below:

"There currently exist two different embodiments of the software according to the present invention. In a first, version of the software, the creation and recipient executable software is loaded into the memory 22B of the sender computer 20. When this embodiment of the software is used, the E-mail message structure contains, as described hereinafter, a portion that attaches the recipient executable software to the E-mail message transmitted from the sender to the recipient. So long as the recipient computer 30 can operate as a Java virtual machine, the recipient computer 30 will receive the E-mail message containing the recipient executable software, preferably written in Java, and use that recipient executable software to display the enhanced E-mail message on the display 38 of the recipient computer 30."

Appellant denies that "it would have been obvious to a person having ordinary skill in the art to modify the SIP: Session Initiation Protocol taught by Handley to include the embedded software code taught by Venkatraman in order to enable the implementation of enhanced services not provided for by the native SIP protocol" as alleged by the Examiner for the following reasons:

1. Venkatraman is concerned only with enhanced email systems and enhanced email messages. It provides absolutely no teaching of application to any other types of message, and certainly no suggestion of applying the teaching of embedding executable software in SIP messages.

2. Email messages and SIP messages are very different types of message as set out below:

   a. Email messages are user to user messages whereas SIP messages are machine to machine. An end user will not normally have visibility of the contents of a SIP message (unlike the voice or video data sent using RTP in a session established using SIP). In contrast, the very purpose of an email message is that it should be seen by the recipient user. (The Examiner has argued against this that email messages are also sent from computer to computer which is, of course, correct, but which completely misses the point. The point is that email messages are intended to be seen and acted on by human users, whereas SIP messages are not);

   b. Email messages provide content (such as text, photos, video etc) to users, whereas SIP messages are signaling messages (ie are used only to manage communications not to actually send content);

   c. Email messages are peer to peer messages, whereas SIP messages are client/server messages. Client/server messages are either request messages sent by a client to a server, or response messages sent by a server to a client. In contrast, peer to peer messages are sent between entity which are peers – ie they are not in an asymmetric relationship of server or client;

d. Email messages exist at Layer 7 (the Applications Layer) of the OSI seven layer model (see table above). SIP messages exist at Layer 5 (the Session Layer).

3. As discussed above, Handley is the industry standard on SIP, yet Handley fails to disclose the possibility of storing computer software code in a SIP message and actually teaches away from so doing - see section 8.1 set out above.

In response to Appellant's earlier arguments that motivation to combine is lacking, the Examiner has argued (see the latest Advisory Action, dated August 29, 2005) that:

"the focus of Venkatraman's system is in fact to create a container that contains executable embedded software to be encrypted in a message [an email message to be precise]. Hence the combination of Handley and Venkatraman would allow for this type of executable embedded software to be applied with the messages of Handley's system, namely SIP messages, thus meeting the limitation of the claim in question." [emphasis added]

However, the issue, as a mater of law, is not whether the prior art "would allow" for the application of features of one reference to the system of another reference. The issue is whether there is motivation in the prior art to modify or combine the teachings of the prior art. The Examiner has completely failed to provide any evidence in the prior art which demonstrates why one skilled in the art would be motivated to apply the teachings of Venkatraman to Handley.

The Examiner has also argued (see the latest Advisory Action, dated August 29, 2005) that:

"both systems [Handley and Venkatraman] are focused on the transfer of multimedia content from one user to another."

Appelant disagrees. This statement is true of Venkatraman but not of Handley. As well known to one skilled in the art and as discussed above, Handley is concerned with establishing, managing and terminating sessions. It is actually entirely independent of the protocol used to transfer content from one user to another, and entirely independent of the type of media to be transferred (see IETF RFC 2543 abstract, set out above).

Second, There is No Reasonable Expectation of Success, and Third, the Prior Art Lacks the Limitations

To establish a *prima facie* case of obviousness, there must be also a reasonable expectation of success that the combination would work. Even if there was motivation in the prior art to make the alleged combination, which is denied, Appellant denies that there would be any reasonable expectation of success.

The teaching in Venkatraman of embedding executable software in an email message is specifically to enable the recipient computer to be able to present enhanced content (which is also embedded in that very same email) to the user when the email arrives. It assumes that the recipient computer does not already have the software required to present the enhanced content. If this is not the case, then no executable software is embedded – see second embodiment of Venkatraman. SIP is a signaling protocol and is independent of protocols (such as RTP) used to actually transfer user content (eg voice, video, multimedia). Even though SIP messages may be used to set up a communications session, there is no guarantee that any actual user data (ie content) will ever be sent.

Since there is no content being sent in the SIP messages of Handley, what would the effect be of embedding the executable software of Venkatraman in a SIP message? There would be no effect. The SIP messages would be received and processed as normal, and the embedded software would be executed, but since the SIP message cannot contain any user content, then there would be nothing for the executed software to present to the user.

Furthermore, as discussed above, SIP messages may also be used to manage existing sessions and to terminate sessions. SIP messages may be request messages such as INVITE, ACK, OPTIONS, BYE, CANCEL and REGISTER messages, and they may be response messages such as 200 OK messages. It is not at all clear what would be the effect of embedding the executable software of Venkatraman in these messages. There is no teaching in Venkatraman of which SIP messages to use (which is not surprising since Venkatraman is concerned only with enhanced email systems and enhanced email messages) nor in Handley (which is again not surprising since Handley specifically teaches away from embedding anything but session description information in SIP messages).

Thus, it is not at all clear that embedding the executable software of Venkatraman in SIP messages would be successful. And, there is nothing in the prior art to teach or suggest the claim limitations initially set forth above.

In view of the above, Appellant believes that the invention as claimed in claim 1 would not have been obvious in view of the prior art references cited by the Examiner. The rejection of claims dependent on claim 1 is moot in view of the above arguments.

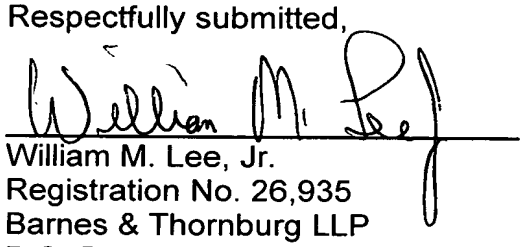
Furthermore, since all other independent claims contain like features (whether storing computer software code in SIP messages or extracting and executing

computer software code from SIP messages) it is submitted that all the claims of the present invention are allowable for reasons similar to those presented above.

Reversal of the Examiner is therefore clearly in order and is solicited.

November 22, 2005

Respectfully submitted,



William M. Lee, Jr.
Registration No. 26,935
Barnes & Thornburg LLP
P.O. Box 2786
Chicago, Illinois 60690-2786
(312) 214-4800
(312) 759-5646 (fax)

Claims Appendix

1. A method of transferring computer software code between a first and a second node in a communications network, each of said nodes comprising a SIP client, said method comprising the steps of:-

(iv) storing computer software code in a SIP message;

(v) sending the SIP message and computer software code from the first SIP client associated with the first node to the second SIP client associated with the second node; and

(vi) executing the computer software using the second node.

2. A method as claimed in claim 1 wherein said computer software code is added to the SIP message.

3. A method as claimed in claim 1 wherein said step of storing computer software code in the SIP message comprises adding an address to the SIP message which indicates where the computer software is stored.

4. A method as claimed in claim 3 wherein said address is a universal resource locator (URL).

5. A method as claimed in claim 1 wherein said computer software code comprises Java byte code.

6. A method as claimed in claim 1 wherein said computer software code comprises one or more Java applets.

7. A method as claimed in claim 1 wherein said computer software code comprises one or more mobile automated software agents.

8. A method as claimed in claim 7 wherein said mobile automated software agents are Java mobile agents.

9. A method as claimed in claim 1 wherein said second node comprises a Java virtual machine.

10. A method as claimed in claim 2 wherein the computer software code is added to the body of the SIP message.

11. A method as claimed in claim 1 which further comprises adding an indicator to a header of the SIP message in order to indicate the presence of the computer software code and arranging the second SIP client to recognise the indicator.

12. A method as claimed in claim 1 which further comprises the step of proceeding with any SIP process related to the SIP message.

13. A method as claimed in claim 11 wherein said second SIP client is arranged such that on receipt of a SIP message containing such an indicator, the computer software code stored in the SIP message is executed by the second node before that second node carries out any other processes related to the SIP message.

14. A method as claimed in claim 1 wherein said computer software is arranged to interact with the second SIP client via a specified application programming interface.

15. A method as claimed in claim 1 wherein said computer software is arranged to interact with a processor associated with the second SIP client via a specified application programming interface.

16. A method as claimed in claim 1 wherein said execution of said computer software code causes the second node to set up a multimedia conference call.

17. A method as claimed in claim 1 wherein said execution of said computer software code causes the second node to upgrade or replace said SIP client.

18. A method as claimed in claim 1 wherein said execution of said computer software code causes the second node to test said second node.

19. A method as claimed in claim 1 wherein said execution of said computer software code causes said second node to collaborate with said first node to forward a call from the first to the second node.

20. A communications network node comprising:

(i) a SIP client;

(ii) an input arranged to receive SIP messages;

(iii) a processor arranged to extract and execute computer software code from a received SIP message.

21. A communications network node as claimed in claim 20 wherein said processor comprises a Java virtual machine.

22. A communications network node as claimed in claim 20 which further comprises an application programming interface arranged to allow the computer software code to interact with the SIP client.

23. A communications network node as claimed in claim 20 wherein said processor further comprises a detector arranged to detect an indicator in a received SIP message which indicates that computer software code is associated with that SIP message.

24. A computer program arranged to control a communications network node, said node comprising a SIP client and a processor, said computer program being arranged to control the node when executed on the processor such that when a SIP message is received by the SIP client, which contains computer software code, the software code is executed by the processor.

25. A computer program as claimed in claim 24 which is stored on a computer readable medium.

26. A communications network comprising a plurality of communications network nodes each such node comprising:

(i) a SIP client;

(ii) an input arranged to receive SIP messages containing computer software code; and

(iii) a processor arranged such that in use, when a SIP message is received, any computer software code contained in that SIP message is executed by the processor.

27. A method of setting up a conference call between two or more parties, each party comprising a SIP client and a host processor, said method comprising the steps of:

(i) storing computer software code in a SIP message;

(ii) sending the SIP message to each of the parties;

(iii) executing the computer software code at each of the host processors.

28. A method as claimed in claim 27 wherein the computer software code is arranged to take into account capabilities of each host processor.

29. A method as claimed in claim 27 wherein said conference call is a multimedia conference call.

30. A system for automatically setting up a conference call between two or more parties, each party comprising a SIP client and a host processor, said system comprising:- a processor for storing computer software code in a SIP message and to send that SIP message to each of the parties; and wherein each of said host processors is arranged to execute the computer software code in use, when the SIP message is received.

31. A method of upgrading or replacing interconnected SIP clients each SIP client being associated with a host processor said method comprising the steps of:-

(i) storing computer software code suitable for said upgrade or replacement in a SIP message;

(ii) sending the SIP message to each of the SIP clients; and

(iii) executing the computer software at each of the host processors.

32. A method of testing members of a group of SIP clients each SIP client being associated with a host processor said method comprising the steps of:-

(i) storing computer software code suitable for said testing in a SIP message;

(ii) sending the SIP message one of the SIP clients;

(ii) executing the computer software at the host processor associated with that SIP client in order to obtain test results; and

(iii) repeating steps (ii) to (iii) for each of the other SIP clients in the group.

33. A method of forwarding a call from a first SIP client to a second SIP client, each of said SIP clients being associated with a host processor, said method comprising the steps of:-

(i) receiving a call at the first SIP client and if that call is not answered then storing computer software code in a SIP message, said computer software code being arranged to forward a call;

(ii) sending the SIP message from the first SIP client to a specified second SIP client; and

(iii) executing the computer software using the host processor associated with the second SIP client such that the call is forwarded to the second SIP client.

Evidence Appendix and Related Proceedings Appendix

There are no such appendices.